Algorithms Lab 2

The in-lab problems are to be solved during the lab time. Work with your team, but write your solutions individually. Your answers will not be graded, however you need to work through these problems; please staple the answers to the in-class exercises to the assignment that you hand in.

The homework problem set is due in one week. Work with your team, but write your solutions individually. List the people with whom you discussed the problems.

Appearance: You are **strongly** encouraged to type your solutions. Please leave plenty of open space for us to write feedback.

1 In lab exercises

1. Finish the recurrence exercises in the handout.

2. Give examples of recurrences that solve to logarithmic time.

3. Give examples of recurrences that solve to linear time.

4. Give examples of recurrences that solve to exponential time.

5. Consider the folowing code for BubbleSort that we discussed in class:

```
BUBBLE-SORT($A$)
1  For $k = 1$ to $n - 1$
2      // do a bubble pass
3      For $i = 0$ to $n - 2$
4          if $A[i] > A[i+1]$: swap
```

(a) Give the best-case and worst-case running time of the algorithm. Express them as $\Theta()$ bounds and give a brief justification.

(b) Show how to change the code so that the algorithm dos not do any redundant bubble-passes (i.e. if the input needs only 3 bubble passes to be sorted, the algorithm does only 3 passes).

2 Homework problems

Your assignment will be evaluated based not only on the final answer, but also on clarity, neatness and attention to details. For example, you'll want to leave plenty of space in between problems so that we can give you feedback.

For problems 1 through 6, find a tight bound for the solution of the following recurrences using iteration.

1. $T(n) = T(n/3) + 1$
2. $T(n) = T(n/3) + n$
3. $T(n) = T(\sqrt{n}) + 1$
4. $T(n) = T(n-1) + n$
5. $T(n) = 7T(n/2) + n^3$
6. $T(n) = 7T(n/2) + n^2$
7. (interview question) (a) Given an unsorted array and a number k. Find two elements in the array whose sum is k, or report if no such set exists. Analyze running time.

   (b) Generalize to 3-sum: Find if there exist 3 elements in the array whose sum is k, or report that no such subset exists. Analyze running time.
8. (interview question, 2014) Suppose you have an n-stories high building, and a bunch of eggs. An egg has a certain level l at which , if thrown from any level $\geq l$, it breaks. For example, an egg might have $l = 7$ meaning you can safely throw the egg down from levels 1 through 6, and it will not break; but if you through the egg from a level 7 or higher, it breaks.

   You are given a building and a bunch of eggs (all identical) and your goal is to find out the level l of the eggs. We can assume $n = 100$ (i.e. 100-level high building).

   (a) Describe an approach that only breaks one egg to find out l. How many throws does it do?

   (b) Describe an approach that minimizes the number of throws. How many eggs might it break?

   (c) Assume now you have two eggs. Describe an approach that minimizes the number of throws.

3 Additional problems

1. (interview question, 2016)

   Input: an array of integer numbers and an integer number k

   Output: if there is a contiguous sequence (subarray) in the given array that sums up to the given number k

   Example [2,1,3,2], k = 4
2. (interview question, 2016) Implement a hash table.